Hide and Speak

Parth Mehta, Swathi Jadav, Srinivasan Sathiamurthy, Vaibhav Agarwal
Carnegie Mellon University, Pittsburgh, PA 15213
parthvim@andrew.cmu.edu, sjadav@andrew.cmu.edu
ssathiam@andrew.cmu.edu, vagarwal2@andrew.cmu.edu

1 Introduction

Our work centers around the foundations laid by [1] that look into deep learning methods for audio steganography. The premise of their research answers the question, "how can one encrypt several hidden messages in a carrier message".

To this end, [1] looked at several ways to effectively hide their messages, and determined that phase modification was the best strategy set among the existing strategies for audio steganography. For these set of strategies, the main issue with decoding the modified fourier-transformed audio signal is aligning the phases across dividers between the time frames being considered: if the phases don't match, then mathematically, the inverse fourier transform doesn't exist. To counter this, they implemented the STFT and ISTFT transforms of the encoder and decoder as differential layers, which effectively acted like a smoother. However, this is still coming at a loss of phase-related information, since such transformations compromise phase related information as well.

They were able to fit up to five hidden messages in a carrier message. They also proposed models that paired keys with the hidden messages.

We wish to extend these results by taking their research question in a cryptographical direction and instead ask, "how can one encrypt several hidden messages in one audio clip with keys such that the user has access to the messages if they input the right key?". Additionally, we wish to extend their phase-modification encoding by injecting both phase-related and magnitude-related information information about the signal frequencies into the encoding and decoding layers instead of the differential STFT/ISTFT layers used in [1].

We currently are implementing models that use one common key for one message, and we hope to generalize that to multiple keys for multiple hidden messages, as well as testing the effectiveness of our alternative phase-modification strategy.

2 Literature Review

Most other prior work has looked into hiding messages in other media, and only hiding up to one message. Audio is an special case in steganography, due to the exceptional amount of redundancies in representation, which gives much more freedom with respect to encoding hidden messages. Additionally, they employed techniques that looked at concealing hidden messages in the phase of frequencies of the carrier message [2], embedding in low amplitude regions [3], and various other transform domains [4]. These methods however generalize poorly for one encoder and decoder across multiple messages, since perturbing a message after its fourier transform and failing to correct the shift before the inverse fourier transform results in audible artifacts, and it is not reasonable to create a single model for this level of exactness across many messages. Thus, [1] concludes that exploitation of specific forms of redundancies is suboptimal. To that end, they propose learning a generalized encryption and decryption model.

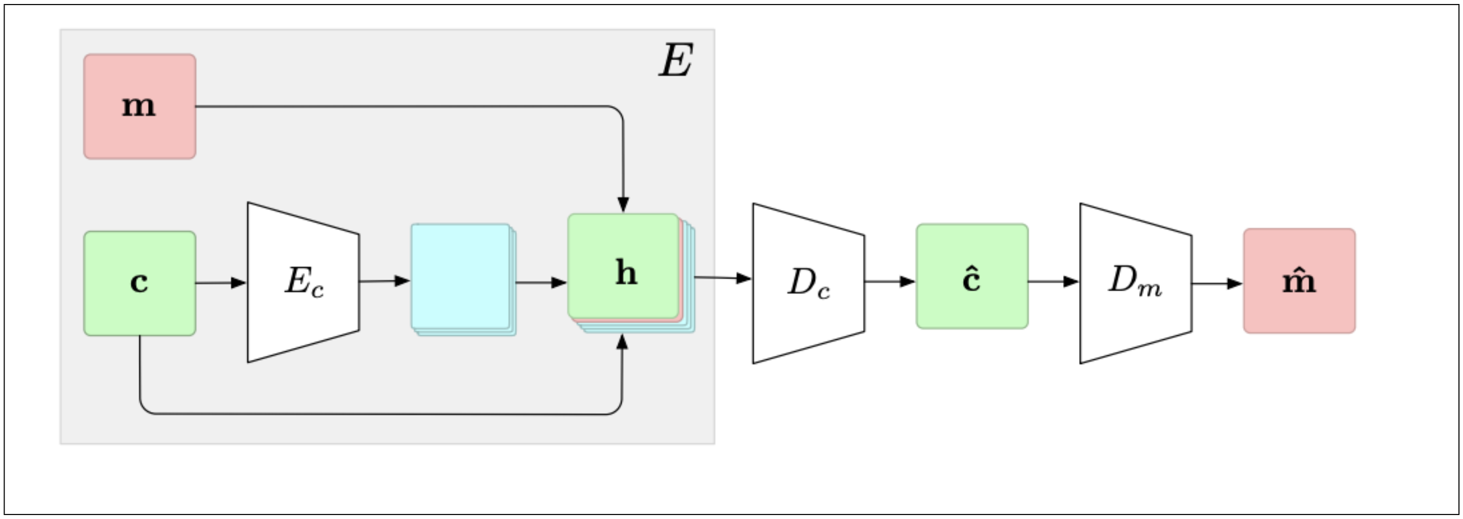

Figure 1: Hide and Speak Model.

In addition to the methods used in [1], our main obstacles which have been previously addressed are data creation, and model structure. For data creation, our problem has the issue of an adversarial decoder (we don't want someone with the wrong key to have access to the messages). To this end, several papers have suggested adding adversarial examples in our data training process, such as messages paired with the wrong key to return the wrong response. This wrong response can be the correct response with slight perturbations, such as in adversarial learning, or more drastic such as minimizing its difference between background noise. They have also suggested using adversarial training examples to encode useful information, such as the keys themselves.

Now for the actual phase modification process of hiding our hidden message in the carrier, there are several methods to do so. (i'll expand on this more tomorrow when I know what's going on) Here's some information:

We defer all discussion about model structures in the next section.

3 Baseline Model Ideas from Research Papers

Our preliminary model will just be a slight modification of the model used in [1]. Refer to figure 1.

In this model, [1] first inputs the carrier message, encodes it, then simply concatenates it with the hidden message(s) m to get h. Then it trains on a decoder network, which takes in the concatenation, and first decodes c (D_c), and then decodes m (D_m). The final loss minimizing function is:

$$\mathcal{L}(c, m) = \lambda_c \cdot \text{MSE}\left(c - D_c(E(c, m))\right) + \lambda_m \cdot \text{MSE}\left(m - D_m(D_c(E(c, m)))\right)$$

where MSE is the mean square error, and λ_c, λ_m are hyperparameters.

We wish to modify this so that we input (m, k_1) instead of just m, and then we input the decoder key k_2 into D_m to get $\hat{m}$ (the decoded message). If k_1, k_2 match, then we want $\hat{m}$ to be minimizied with m. If k_1, k_2 do not match, we want $\hat{m}$ to be minimized with noise. We can add another input after $\hat{m}$, which is concatenated to our training examples, which is equal to m if the keys match, and noise if the keys don't match. The end layer of our model will train on this difference at the end, but we will extract our encoder and decoder from the intermediate layers of the model.

This original model has a carrier message, and hidden messages with keys. In order to just include the just the messages with keys, we can ignore the initial input of m (the first pink square), and change c to the message and key pairs. In summary, here is our preliminary model:

Our data examples will be batches of messages depending on however many messages we want to encode in one audio clip at the same time. So up to three tuples that look like (m_i, k_i, k_i, m_i) or $(m_i, k_i, k_j, \text{noise})$.

- STFT all audio.
- Input the messages through a few CNN layers for feature selection
- Concatenate the messages with their keys (input the keys). This yields h. Run this through a few CNN layers. (you want to keep the keys associated with the messages, but reduce the dimensionality of the concatenated encodings of messages to the size of the original audio clip to combine it into one message)
- Then plug everything into D_m. At this point, input the second key of the training tuples into the model. Run this for a few CNN layers.
- Then compute the loss function (MSE) with the fourth parameter in the tuple (what the resulting message should be)

4 Dataset Description

As mentioned above, our preliminary dataset will consist of training examples that are 4 tuples of the form: (m_i, k_i, k_i, m_i) or $(m_i, k_i, k_j, \text{noise})$, where m_i are hidden messages, and $k_i \neq k_j$ are keys. We picked 3 since more than that made some of the hidden messages incomprehensible.

We will make this data using the TIMIT and YOHO data sets used in [1], and the noise will be random snippets of the appropriate size of one large background restaurant noise file that we will ask Professor Raj of [1] for when the time comes. (Or just pick one ourselves). The way we make each tuple will be to randomly select four audio clips of a specific size from those data sets, which will be our hidden messages. Then we will pick random large numbers (number of bits equal to the length of the audio file), for our first keys. The second keys will be the same with $\frac{1}{2}$ probability, and if they differ, they will be another random large number with the number of bits equal to the length of the audio file. And if the keys are the same, we put the original message as the fourth item in the tuple. If they are different, we will select a random subclip of the noise audio of appropriate size. We will use these data sets via standard test/train/val splits.

We now go into further detail for the YOHO and TIMIT datasets:

4.1 YOHO

YOHO Dataset is provided by the Linguistic Data Consortium (LDC) to enable research, spark competition, and provide a means for comparative performance assessments between various voice verification systems.

The YOHO voice verification corpus was collected while testing ITT's prototype speaker verification system .in an office environment [1]. This database is the largest supervised speaker verification database known to the author.

The speech data is divided into two directories to separate enrollment and verification sessions.

4.2 TIMIT

The NIST Speech Dataset contains the complete Texas Instruments/Massachusetts Institute of Technology (TIMIT) acoustic-phonetic corpus of read speech. TIMIT was designed to provide speech data for the acquisition of acoustic-phonetic knowledge and for the development and evaluation of automatic speech recognition systems.

TIMIT contains a total of 6300 utterances, 10 sentences spoken by each of 630 speakers from 8 major dialect regions of the United States. 70% of the speakers are male and 30% are female.

The 10 sentences represent roughly 30 seconds of speech material per speaker. In total, the corpus contains approximately 5 hours of speech. All speakers are native speakers of American English and were judged by a professional speech pathologist to have no clinical speech pathologies. Recordings were made in a noise-isolated recording booth at TI.The speech was directly digitized at a sample rate of 20 kHz using a Digital Sound Corporation DSC 200 with the anti-aliasing filter at 10 kHz. The speech was then digitally filtered, debiased, and downsampled to 16 kHz.

All of the recorded sentences were provided with a tune- aligned sequence of acoustic-phonetic labels. The label set is intended to represent a level somewhat intermediate between phonemic and acoustic.

5 Baseline Selection

We create a baseline by evaluating the TIMIT dataset based on the existing implementation in [Bhiksha's paper]. This model uses a random sampling to create a pair of audio samples to be used as a carrier and the message, this ensures completely arbitrary message and carrier. Each audio file is sampled at 16kHz and the STFT is applied with W = 256 FFT frequency bins and sliding window with a shift L = 128. The baseline models are trained using Adam optimizer for 80 epochs with an initial learning rate of 103. We balanced between the carrier and message reconstruction losses by using c = 3, m = 1 in the loss function. The baseline we select for our model in the task of concealing a single message is as follows:

- Absolute Error Carrier loss = 0.0016
- Absolute Error Message loss = 0.035

6 Baseline Implementation

Refer to the code in this drive, which we obtained from the authors of [1] and modified to our needs. Comments are provided as well in the code.

Link: https://drive.google.com/drive/folders/19NhdfIIG2Wl9Sac1LpXGDxlrMs5WybRf?usp=sharing

7 Proposed Extensions

We still need to implement the multiple keys as well as our alternative phase modification inputs (which will just be more outputs/inputs into the layers after the STFT and before the ISTFT): specifically, the outputs from the STFT and the inputs into the ISTFT. We hope that this will capture the phase-alignment information better than just making those two layers differentiable and training on closeness to the carrier and hidden messages.

8 Acknowledgments

We thank our mentor, Manasi Purohit, and our professor, Dr. Raj, for providing us the project idea and guiding us through the research process.

References

[1] Kreuk, Adi, Raj, Singh, and Keshet.Hide and Speak: Deep Neural Networks for Speech Steganography. In ReseachGate (2019)

[2] Dong, X., Bocko, M. F., and Ignjatovic, Z. Data hiding via phase manipulation of audio signals. In Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on, volume 5, pp. V–377. IEEE, 2004.

[3] Shirali-Shahreza, S. and Shirali-Shahreza, M. Steganography in silence intervals of speech. In International conference on intelligent information hiding and multimedia signal processing, pp. 605–607. IEEE, 2008.

[4] Djebbar, F., Ayad, B., Meraim, K. A., and Hamam, H. Comparative study of digital audio steganography techniques. EURASIP Journal on Audio, Speech, and Music Processing, 2012(1):25, 2012.

More papers to be added tomorrow